EXPERIMENT - 33

Object Oriented Programming Lab

Aim

Write a program to read a text file and display its contents on the screen.

Syeda Reeha Quasar
14114802719
4C7

EXPERIMENT – 33

Aim:

Write a program to read a text file and display its contents on the screen.

Source Code:

```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;

int main () {
  // Create a text file
  ofstream MyWriteFile("filename.txt");

  // Write to the file
  MyWriteFile << "Files can be tricky, but it is fun enough! This is a file writt
en and we are reading this!! \n \n some figures too 123 \n\n";

  // Close the file
  MyWriteFile.close();

  // Create a text string, which is used to output the text file
  string myText;

  // Read from the text file
  ifstream MyReadFile("filename.txt");

  // Use a while loop together with the getline() function to read the file line
by line
  while (getline (MyReadFile, myText)) {
    // Output the text from the file
    cout << myText;
  }

  // Close the file
  MyReadFile.close();
}
```

Output:

```
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ read.cpp -o read } ; if ($?) { .\read }
Files can be tricky, but it is fun enough! This is a file written and we are reading this!!   some figures too
123
PS D:\sem 4\cpp\oops>
```

```
Files can be tricky, but it is fun enough! This is a file written and we are reading this!!   some figures too
123
```

```
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ read.cpp -o read } ; if ($?) { .\read }
Files can be tricky, but it is fun enough!
```

Viva Questions

Q1). What is file handling in C++?

Files store data permanently in a storage device. With file handling, the output from a program can be stored in a file. Various operations can be performed on the data while in the file.

A stream is an abstraction of a device where input/output operations are performed. You can represent a stream as either a destination or a source of characters of indefinite length.

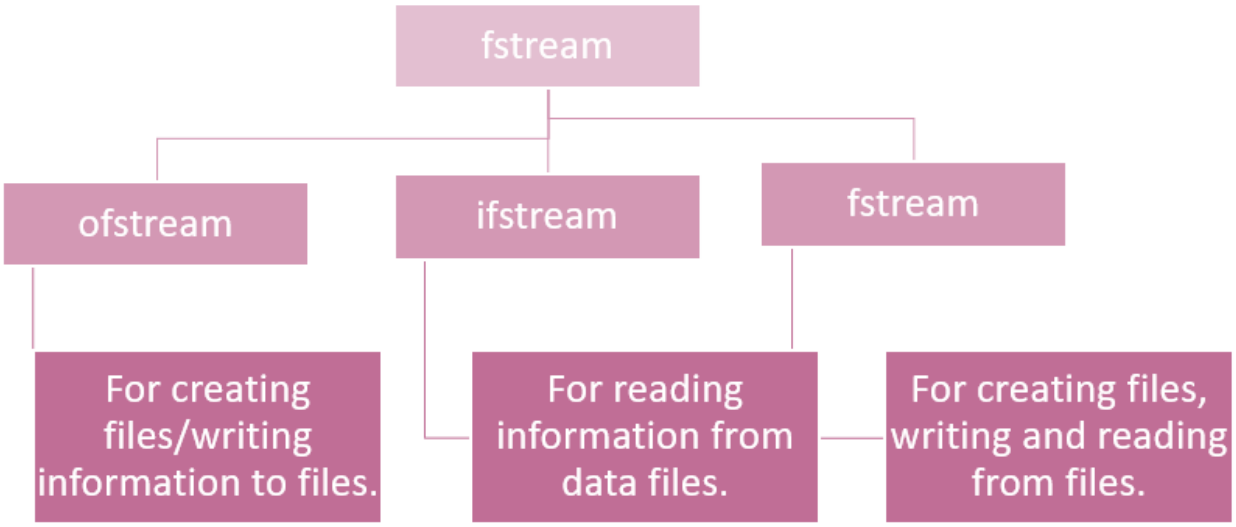
Q2). What is fstream Library?

Ans.

The fstream library provides C++ programmers with three classes for working with files. These classes include:

- **ofstream**- This class represents an output stream. It's used for creating files and writing information to files.
- **ifstream**- This class represents an input stream. It's used for reading information from data files.
- **fstream**- This class generally represents a file stream. It comes with ofstream/ifstream capabilities. This means it's capable of creating files, writing to files, reading from data files.

The following image makes it simple to understand:



fstream library

To use the above classes of the fstream library, you must include it in your program as a header file. Of course, you will use the #include preprocessor directive. You must also include the iostream header file.

Q3). What are benefits of file handing?

Ans.

- With file handling, the output of a program can be sent and stored in a file.
- A number of operations can then be applied to the data while in the file.
- A stream is an abstraction that represents a device where input/output operations are performed.
- A stream can be represented as either destination or source of characters of indefinite length.
- The fstream library provides C++ programmers with methods for file handling.
- To use the library, you must include it in your program using the #include preprocessor directive.

Q4). What are the benefits of OOPs?

Ans.

The procedural-oriented languages focus on procedures, with function as the basic unit. You need to first figure out all the functions and then think about how to represent data.

The object-oriented languages focus on components that the user perceives, with objects as the basic unit. You figure out all the objects by putting all the data and operations that describe the user's interaction with the data.

Object-Oriented technology has many benefits:

- *Ease in software design* as you could think in the problem space rather than the machine's bits and bytes. You are dealing with high-level concepts and abstractions. Ease in design leads to more productive software development.
- *Ease in software maintenance*: object-oriented software are easier to understand, therefore easier to test, debug, and maintain.
- *Reusable software*: you don't need to keep re-inventing the wheels and re-write the same functions for different situations. The fastest and safest way of developing a new application is to reuse existing codes - fully tested and proven codes.